

Fuzzy User Profile Intrusion Detection Systems (FUPIDS) and Kernel Service Profile Intrusion Detection Systems (KSPIDS)

Steffen Wendzel
steffenwendzel (at) gmx (dot) net
<http://www.wendzel.de>

May-22-2008, Version 1.1

Contents

1	Introduction to User Profile IDS and FUPIDS	1
2	Current Implementations	2
2.1	KSPIDS	2
2.2	FUPIDS2 and Human oriented IDS	3
2.3	NNID	3
2.3.1	Differences between FUPIDS to NNID	3
2.4	PFACollection	4

Abstract

This paper shows two very similar user program execution based intrusion detection ideas I implemented within the last years. It was written to share the ideas and includes no qualitative tests. All IDS ideas are available as proof of concept implementations. Such intrusion detection systems exist since years and this paper will also mention one of the most popular similar systems (NNID). Another similar IDS technology I call human oriented IDS is also described.

1 Introduction to User Profile IDS and FUPIDS

The main idea of User Profile IDS is to modify the `exec*()` syscall implementation within the kernel (FUPIDS1).¹ The modification is made to register all

¹FUPIDS2 tried to do that within userspace by permanently checking the list of running programs, what is very bad since it does not allow a permanent monitoring and it is not very safe since one can't make sure that all executed programs will be caught.

exec*() events of all users. This makes it possible to store all information about executed programs of every user.

FUPIDS basically uses a list of all users and their used programs and a counter that shows how many executions of a program were made. The intrusion detection is based on this information. Imagine a user that uses vi, ls, bash, w, uptime and some other programs every day. FUPIDS will register all these programs for this user. Now imagine that this user now starts using some strange new programs (he for example changes his editor to elvis and then runs something like gcc, ld, cc1, as, perl, ./exploit_xyz and the like). What FUPIDS does is to register all these events and rises an attacker level for the user since he usually doesn't make use of these new programs. It will then print out an log message telling the administrator about these new programs and the current attacker level of the user (the attacker level will probably be "HIGH").

How it works: An very unusual behavior rises the attacker level more than an usual behavior. And an very usual behavior shrinks the attacker level. This means: If the user uses vi, bash, ls and the like every day, his attacker level will be very low, maybe even zero. If he executes a new program, the attacker level will rise. If he executes the new program a second time, the attacker level will rise too (but not so high like if this program was executed for the first time). If he executes the program for the 20th time the attacker level will rise only a bit.

2 Current Implementations

I developed three different proof of concept ideas:

- User Profile Intrusion Detection Systems (UPIDS)
 - FUPIDS 1 (0.0.4) as an OpenBSD Kernel Patch (PoC) in 2003
- Human Oriented Intrusion Detection Systems (HoIDS)
 - FUPIDS 2 as an Userspace version of FUPIDS 1 (PoC) including so called human oriented IDS features in 2005 based on a simple neural feed forwarding network
- Kernel Service Profile Intrusion Detection Systems (KSPIDS)
 - KSPIDS 0.0.3-2008 as an Linux 2.6 Kernel Patch (PoC) in 2008

2.1 KSPIDS

I already described FUPIDS within the introduction and in [WEND03]. Now I want to describe KSPIDS.

While FUPIDS is specialized in monitoring normal system users on an OpenBSD system KSPIDS is specialized in monitoring service users (e.g. www-data) on a Linux system. Since service users only use a very few number of programs, the calculation of an attacker level of such an users is much easier

(e.g. a normal user could change his favorite editor but the service user will not change it).

Because of the different user ranges FUPIDS by default monitors users with an ID bigger than 999 while KSPIDS monitors UIDs between 1 and 999.

More information about KSPIDS can be found in [WEND08].

2.2 FUPIDS2 and Human oriented IDS

FUPIDS2 is a user space version of FUPIDS adding features of human oriented IDS (HoIDS) to it. It uses a simple neural feed forwarding network and its primary field of application are huge workstation pools (e.g. at a campus area network).

What I call HoIDS is collecting and using indirect available physical user data one can get to monitor user specific behavior. Additional data compared with FUPIDS is physical behavior (e.g. Does the user prefer to sit near the window or near a door? Does he prefer to sit in front part of a room? In which building in which room does he prefer to use his computer? At which time at which day of the week does he use which programs on which computers?).

FUPIDS2 was not made to monitor everything a user does. It only monitors data configured by the administrator (the administrator has to configure the location of each workstation to monitor). FUPIDS2 is in theory a pretty interesting technology but in practice such a system can be used for heavy user monitoring what could end in misuse.

More about FUPIDS2 can be found in [WEND05]

2.3 NNID

There are different implementations like mine and some of them even existed years before I invented FUPIDS. A similar implementation was made by Lain, Ryan and Miikkulainen at CS department at the University of Texas [LRMIDNN]. They developed a system called NNID:

NNID is a backpropagation neural network trained to identify users based on what commands they use during a day. The system administrator runs NNID at the end of each day to see if the users sessions match their normal pattern. If not, an investigation can be launched. [LRMIDNN, p. 2]

2.3.1 Differences between FUPIDS to NNID

The big differences between NNID and FUPIDS seem to be the following (I do not have access to NNID and can only use the information I found in the paper):

- NNID is based on a neural network, FUPIDS uses very simple fuzzy logic. The advantage of FUPIDS that there is no training phase needed.
- NNID was implemented in NetBSD while FUPIDS1 used OpenBSD

- NNID was limited to the 100 most common used programs while FUPIDS does not depend on a pre-defined list of programs to monitor (it can monitor all programs)
- NNID also seems to check if a users daily profile matches his usual behavior. FUPIDS instead checks the behavior in realtime and also alerts in realtime.
- FUPIDS (instead of KSPIDS) provides some special additional features used to calculate the attacker level of an user: it also uses promiscuous mode detection and monitors listen() syscalls.

2.4 PFACollection

There is also an interesting tool developed by Seth Freeman et al. It is based on the Solaris Basic Security Module (BSM) log data and creates an probabilistic finite automata that represents the program using behavior (including information about the order the programs where executed) for every user. More information about PFACollection can be found in [SFREE02].

References

- LRMIDNN: Lian, Ryan, Miikkulainen: Intrusion Detection with Neural Networks, Departement of Computer Science (University of Texas), May-1998.

Bibliography

- SFREE02: Seth Freeman: Host-Based Intrusion Detection Using User Signatures, May-2002.
- WEND03: Steffen Wendzel: Implementierung eines Fuzzy User Profile IDS in den OpenBSD-Kernel”, <http://files.doomed-reality.org/Papers/fupids.pdf>, Dec-2003.
- WEND05: Steffen Wendzel: Human oriented IDS & FUPIDS 2, http://files.doomed-reality.org/Papers/hoids_presentation.pdf, Dec-2005.
- WEND08: Steffen Wendzel: KSPIDS Overview, <http://wendzel.de/?sub=softw&ssub=kspids>, May-2008.